Pacific Community
Communauté du Pacifique

Digital Earth PACIFIC

DEPAL:

Abstraction Library Introduction and Walk-through

Sachindra Singh
Geoinformatics Team Leader
Pacific Community (SPC)
sachindras@spc.int

Introduction

Using Python for data science applications such as Earth Observations is the standard in academia and industry.

While the Python Language is relatively simple, and easy to learn, Earth Observation analysis uses large number of libraries which can be difficult and time-intensive to learn.

To alleviate and expedite capacity building for non-programmers and intermediate GIS technicians, DEP has implemented the **DEP Abstraction Library (DEPAL)** into the computing hub environment, which enables users to quickly undertake common tasks with:

- **Fewer lines of codes**
- **Sane defaults**
- **Pacific-oriented**
- **Lower Barrier to Entry**
- **Easy to use and learn**

Capabilities

Some examples of the functionalities provisioned by the abstraction library includes:

- Defining “area of interest” (AOI) by Pacific Island Countries national, maritime and administrative boundaries,
- Enabling using custom “area of interest” (AOI)
- Rapidly querying satellite data catalog on images and banding,
- Creating cloudless images
- Undertaking multi spectral analysis e.g.: for vegetation and crops
- Visualising and analysing global Landcover models
- Support functions for Landcover Generation

DEP will continue to extend and simplify the abstraction library based on end-user feedback and enhance it's built-in functionalities.

Recap: Initial Setup

- Login to https://dep-staging.westeurope.cloudapp.azure.com/ with your provisioned access.
- Open terminal by choosing on the menu: **File, New -> Terminal**
- Clone the DEPAL library repository by copying and pasting one-by-one the following commands and pressing enter:

```
cd
git clone https://github.com/digitalearthpacific/depal
cd depal
```

Recap: Reference Documentation

Documentation of functions would be under <u>depal/doc subfolder</u> named **depal.pdf** in your JupyterHub file browser, and available in print table.

depal (version 0.0.1)

index
/home/sachin/Projects/depal/depal.py

depal.py: Digital Earth Pacific (Abstration Library)

Modules

cartopy.crs	xrspatial.multispectral	matplotlib.pyplot	
geopandas	numpy	pystac_client	
itertools	planetary_computer	rasterio	stackstac
matplotlib	pandas	rioxarray	xarray

Functions

chart_global_land_cover(data)
    # Annual Charting of Land Cover Classes

cleanup()
    # Cleanup Dask Resources

colour_maps()
    # List Colour Maps

do_coastal_clip(aoi, data, buffer=0)
    # Clip Coastal Buffer (by 100 Metres Intervals)

get_area_from_geojson(geojson_file)
    # AOI from GeoJson File (use geojson.io)

get_cloudless_mosaic(aoi, collection_name='sentinel-2-l2a', timeframe='2019-11-01/2022-11-31', cloudcover=10, resolution=100, max=100, period='yearly', coastal_clip=False)
    # median composite - Cloudless Mosaic achieved combining images across time

get_country_admin_boundary(country, admin_type, admin)
    # AOI from Country Administrative Boundary

get_country_boundary(country)
    # AOI from a Country Nation Boundary

get_data(aoi, bands=[], collection_name='sentinel-2-l2a', timeframe='2023-01-01/2023-12-31', cloudcover=10, resolution=100, max=<built-in function max>, period='monthly', coastal_clip=False)
    # Xarray Dataset from STAC by Period Yearly, Quarterly, Monthly, Weekly, Daily

get_evi(aoi, collection_name='sentinel-2-l2a', timeframe='2019-11-01/2022-11-31', cloudcover=10, resolution=100, max=100, period='monthly', coastal_clip=False)
    # evi - Enhanced Vegetation

get_gci(aoi, collection_name='sentinel-2-l2a', timeframe='2019-11-01/2022-11-31', cloudcover=10, resolution=100, max=100, period='monthly', coastal_clip=False)
    # gci - Green Chlorophyll Index

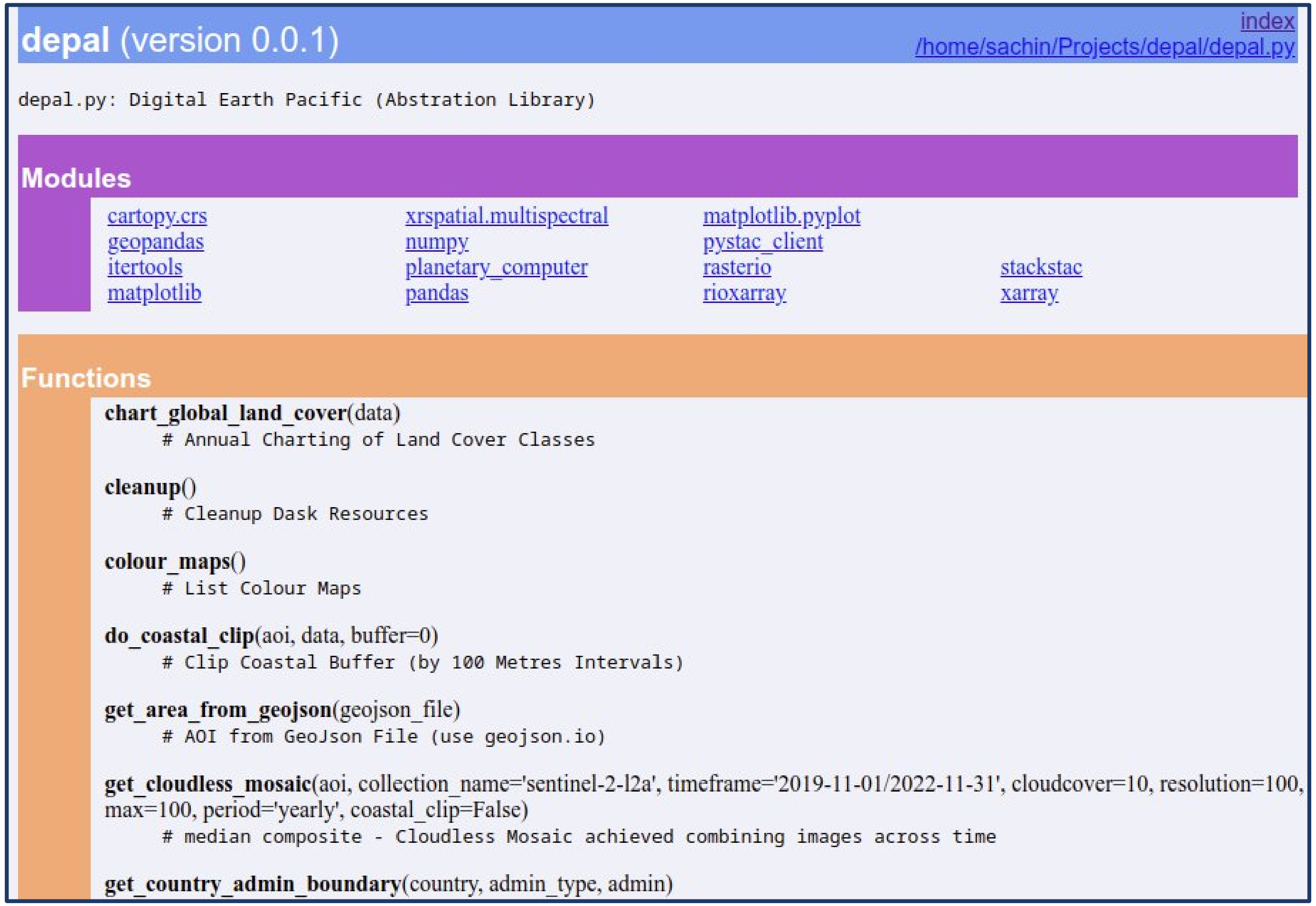
get_global_land_cover(aoi, name='io-lulc-9-class')
    # Get Global LandCover over AOI

get_landcover_mosaic(aoi, year, bands=['B02', 'B03', 'B04', 'B05', 'B06', 'B07', 'B08', 'B8A'], resolution=10, max=10000, cloudcover=10, collection_name='sentinel-2-l2a', coastal_clip=False)
    # Generate Annual Landcover Mosaic with Multiple Bands for ML Classification

get_latest_images(aoi, collection_name='sentinel-2-l2a', timeframe='2023-01-01/2023-12-31', cloudcover=10, resolution=100, max=100, period='daily', coastal_clip=False)
    # Latest RGB Images

get_ndmi(aoi, collection_name='sentinel-2-l2a', timeframe='2019-11-01/2022-11-31', cloudcover=10, resolution=100, max=100, period='monthly', coastal_clip=False)
    # ndmi - Normalised Difference Moisture Index

file:///home/sachin/Projects/depal/doc/depal.html 1/3



depal (version 0.0.1)

index
/home/sachin/Projects/depal/depal.py

depal.py: Digital Earth Pacific (Abstration Library)

Modules

cartopy.crs xrspatial.multispectral matplotlib.pyplot
geopandas numpy pystac_client
itertools planetary_computer rasterio stackstac
matplotlib pandas rioxarray xarray

Functions

chart_global_land_cover(data)
    # Annual Charting of Land Cover Classes

cleanup()
    # Cleanup Dask Resources

colour_maps()
    # List Colour Maps

do_coastal_clip(aoi, data, buffer=0)
    # Clip Coastal Buffer (by 100 Metres Intervals)

get_area_from_geojson(geojson_file)
    # AOI from GeoJson File (use geojson.io)

get_cloudless_mosaic(aoi, collection_name='sentinel-2-l2a', timeframe='2019-11-01/2022-11-31', cloudcover=10, resolution=100, max=100, period='yearly', coastal_clip=False)
    # median composite - Cloudless Mosaic achieved combining images across time

get_country_admin_boundary(country, admin_type, admin)

Updating

DEPAL is frequently updated and enhanced with additional functionalities. Training activities and walkthroughs are also updated and uploaded.

It is a good practice to update DEPAL frequently, by following the steps below:

1. Open terminal clicking File, New -> Terminal.

2. Change into the depal folder by

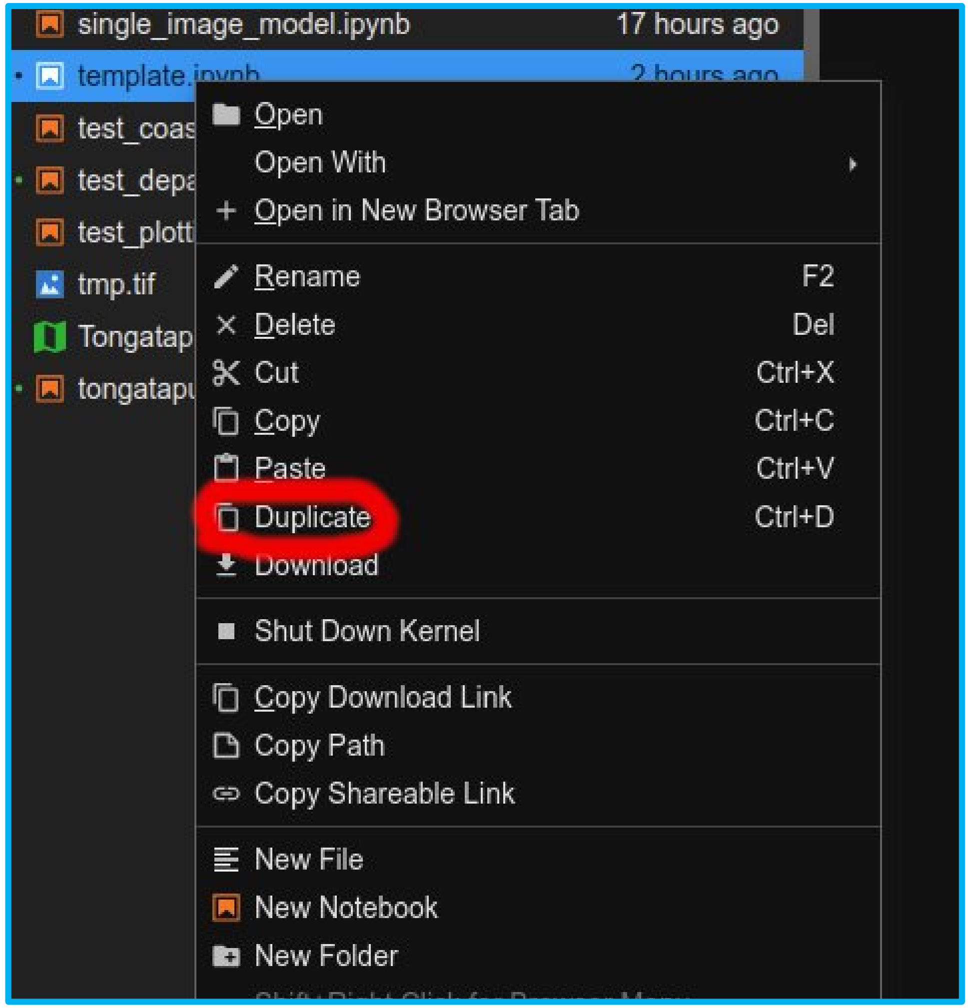
```
cd
```

```
cd depal
```

3. Get latest changes, documentation and updated training materials by inputting the following in the terminal:

```
git pull
```

Starting a new DEPAL Notebook



Under depal folder in JupyterHub, right click **template.ipynb** and select Duplicate.

A new notebook will be created named *Untitled.ipnyb*. This has all defaults pre-populated.

Right click Untitled.inyb and Rename it your *name*.ipynb eg: *sioeli.ipynb*

Edit and add new cells, by clicking the + button only between `dep.init()` and `dep.cleanup()` cells.

Recap: Defining Area of Interest (1)

Defining AOI is a prerequisite to undertaking Earth Observation analysis.

DEPAL allows users to define AOI by regional administrative boundaries per country

Walkthrough:

Listing Administrative Boundary Types

```
dep.list_boundary_types(“Tonga”)
```

```
dep.list_country_boundary(“Tonga”, “Island Group”)
```

Get Country Boundary by Admin Type eg: Island, District

```
aoi = dep.get_country_admin_boundary(“Tonga”, “Island Group”, “Tongatapu”)
```

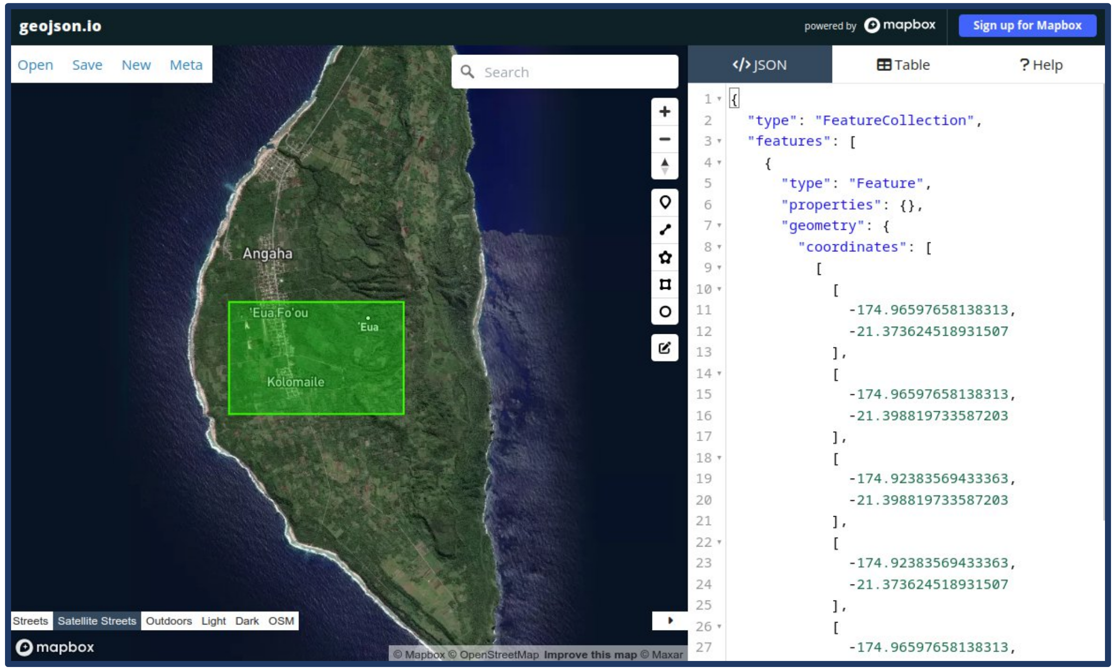
Visualise AOI

```
aoi.explore()
```

Recap: Defining Custom Area of Interest (2)

Walkthrough

- Open **geojson.io** in your browser.
- Zoom to your area of interest
- Draw a polygon.
- Copy the generated JSON from the right hand side box.
- Create a new text File In JupyterHub.
- Copy and paste the JSON data from above in to the file.
- Save the file as custom.geojson
- In your notebook, load and visualise the AOI using:




```
aoi = dep.get_area_from_geojson("custom.geojson")
aoi.explore()
```

Getting Latest Images

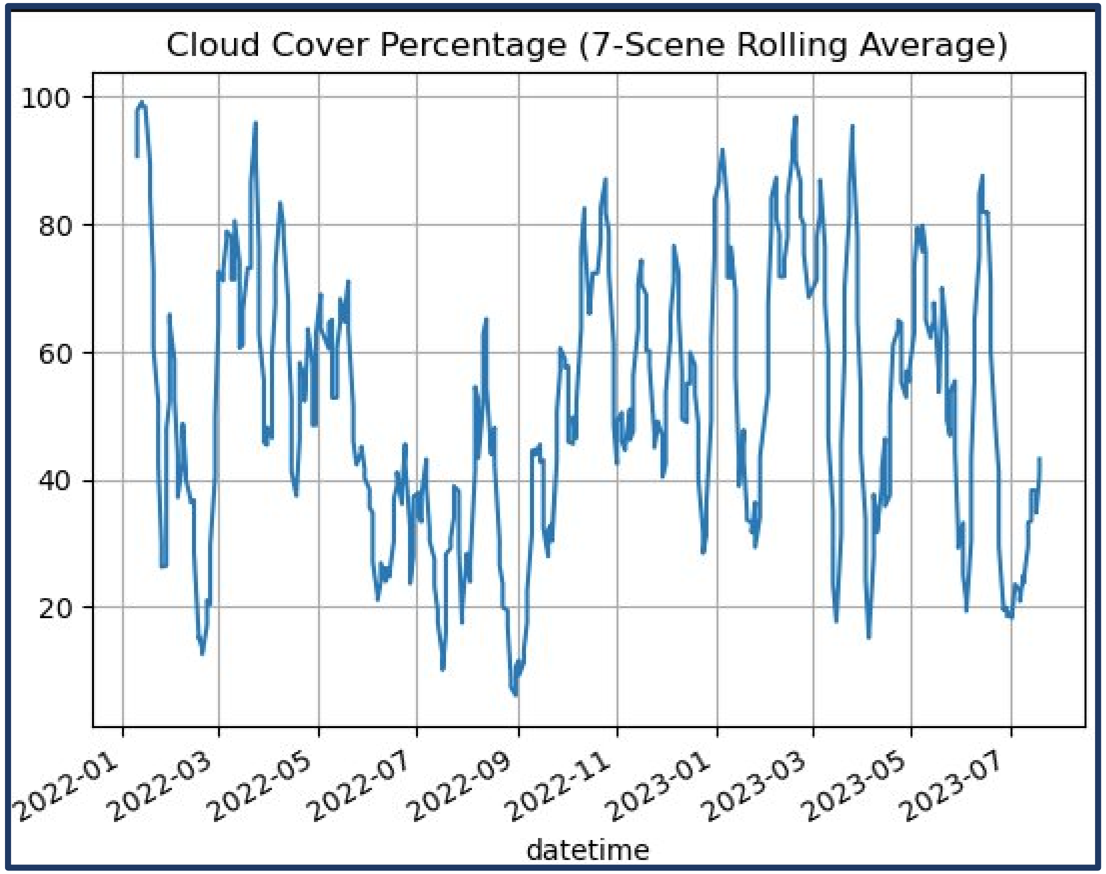
```
data = dep.get_latest_images(aoi, timeframe="2022-08-31/2022-12-31")
```

```
dep.visualise(data)
```

Note prevalence of cloud cover and incomplete full capture over AOI.

Plotting Cloudiness

```
dep.plot_cloudiness(aoi, timeframe="2022/2023")
```

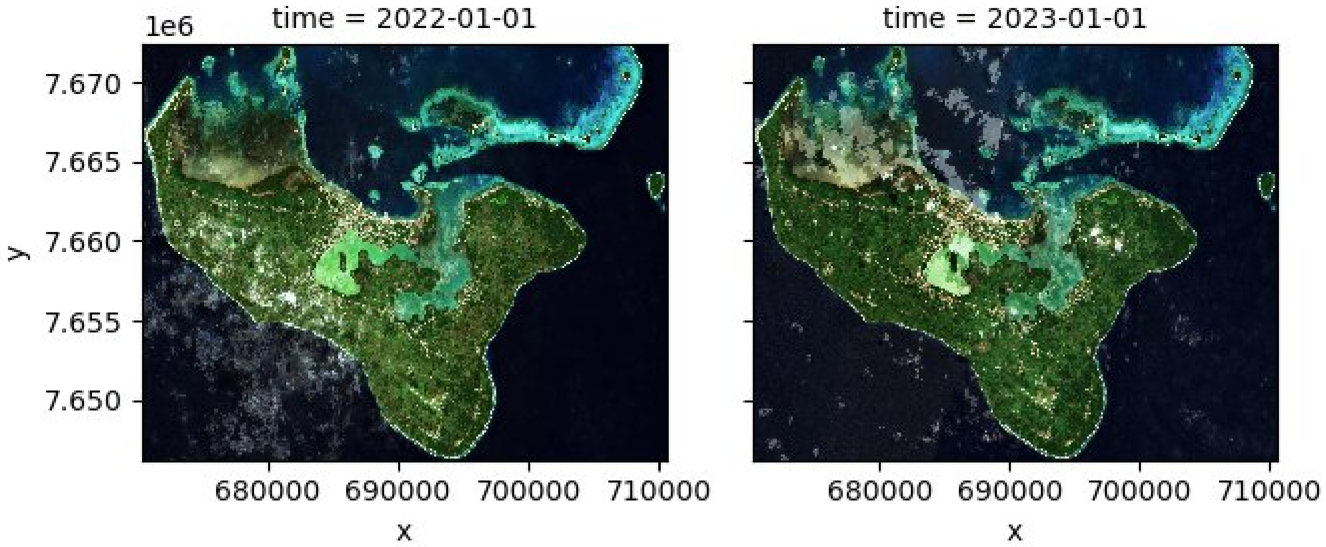


Generate Annual Cloudless Mosaic

To mitigate cloud cover and data gap issues, EO products primarily work on cloudless mosaics across a time period, mostly annually.

```
data = dep.get_cloudless_mosaic(aoi, timeframe="2022/2023", period="yearly")
```

```
dep.visualise(data)
```



DEPAL Default Parameters and Options

Parameter	Description	Default	Options
resolution	Image resolution of output product in metres/square	100	10 - 500
cloudcover	Minimum percentage of cloud cover over source imagery	10	0-100
coastal_clip	Remove coastal, ocean areas	False	True, False
period	Analysis period of products	monthly	yearly, quarterly, monthly, weekly, daily
max	Maximum number of images/scene to perform analysis on	100	1-10000
timeframe		None	*Examples:* 2023 2022/2023 2023-01-30/2023-06-30
collection_name	Satellite constellation/sensor to use	sentinel-2-l2a	sentinel-2-l2a, landsat-c2-l2

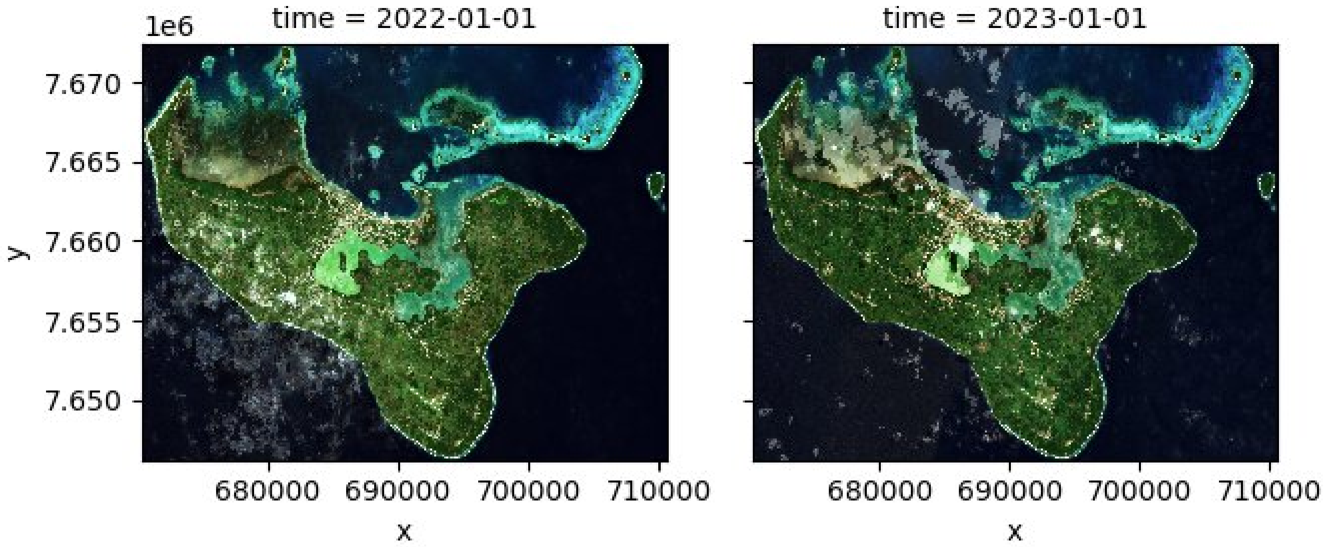
```
data = get_ndmi(aoi, collection_name="sentinel-2-l2a", timeframe="2019-11-01/2022-11-31", cloudcover=10,  resolution=default_resolution, max=default_max, period="monthly", coastal_clip=False):
```

Generate High-Res Annual Cloudless Mosaic

Generate highest resolution cloudless mosiac for downloading.

```
data = dep.get_cloudless_mosaic(aoi, timeframe="2022/2023", period="yearly", resolution=10)

dep.visualise(data)
```

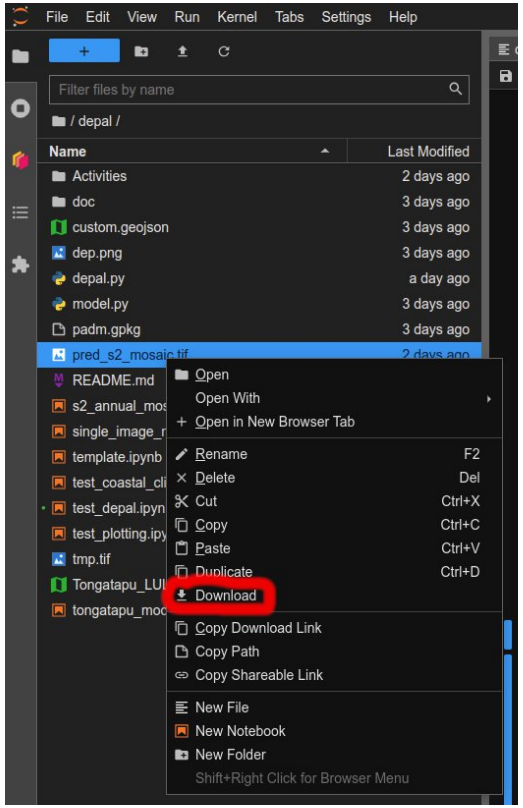


Saving and Downloading Outputs

`dep.save_multiple(data, “cloudless”)` *You can specify any file name here*

This will save *cloudless_1.tif* and *cloudless_2.tif files* each for the respective years of 2022 and 2023.

In JupyterHub the files can be right-clicked and downloaded for visualisation and other analysis within QGIS.



If you have generated annual mosaic for only a single year eg: 2016, you can save single image by:

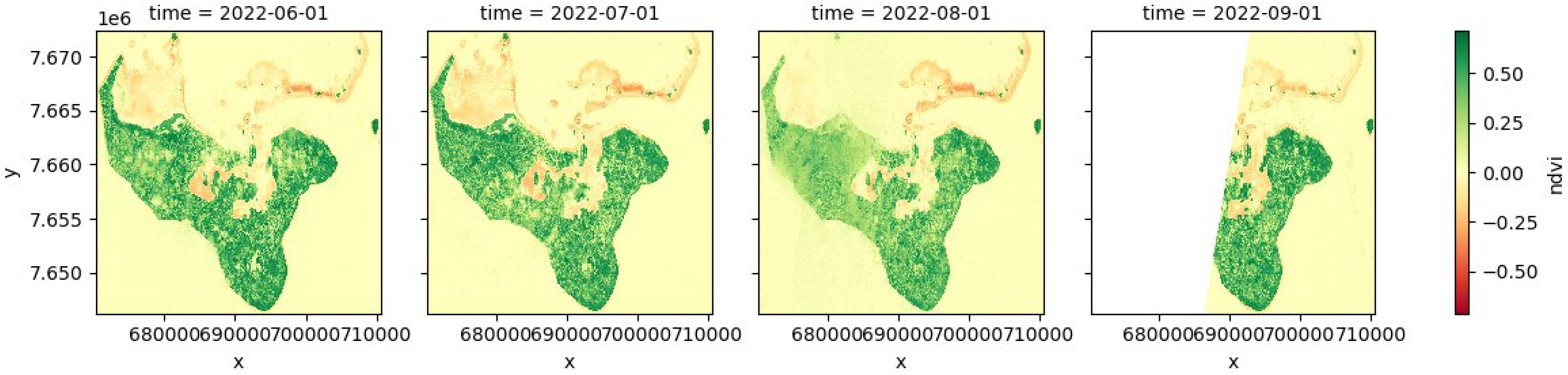
`dep.save_single(data, "s2_2016")`

Generate Normalised Difference Vegetation Index

The **normalized difference vegetation index (NDVI)** is a widely-used metric for quantifying the health and density of vegetation using sensor data.

```
data = dep.get_ndvi(aoi, timeframe="2023-01-01/2023-07-30", period="monthly")

dep.visualise(data, cmap=”RdYlGn”)
```



Generate Enhanced Vegetation Index

Computes **Enhanced Vegetation Index** (EVI). Allows for improved sensitivity in high biomass regions, de-coupling of the canopy background signal and reduction of atmospheric influences. Low-laying vegetation.

Used to identify stress related to drought over different landscapes. Mainly associated with the development of droughts affecting agriculture.

```
data = dep.get_evi(aoi, timeframe="2022-06-01/2023-07-30", period="monthly")
```

```
dep.visualise(data, cmap=”RdYlGn”)
```

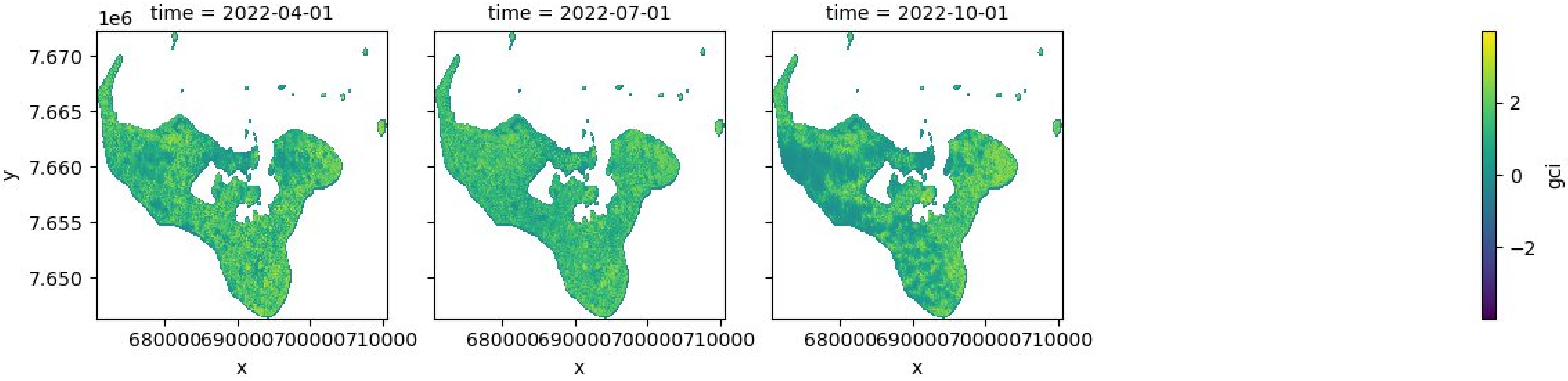
Generate Green Chlorophyll Index (Crop Health)

Computes **Green Chlorophyll Index (GCI)**.

Used to estimate the content of leaf chorophyll and predict the physiological state of vegetation and plant health.

```
data = dep.get_gci(aoi, timeframe="2022-06-01/2023-07-30", period="monthly")
```

```
dep.visualise(data, cmap=”viridis”)
```



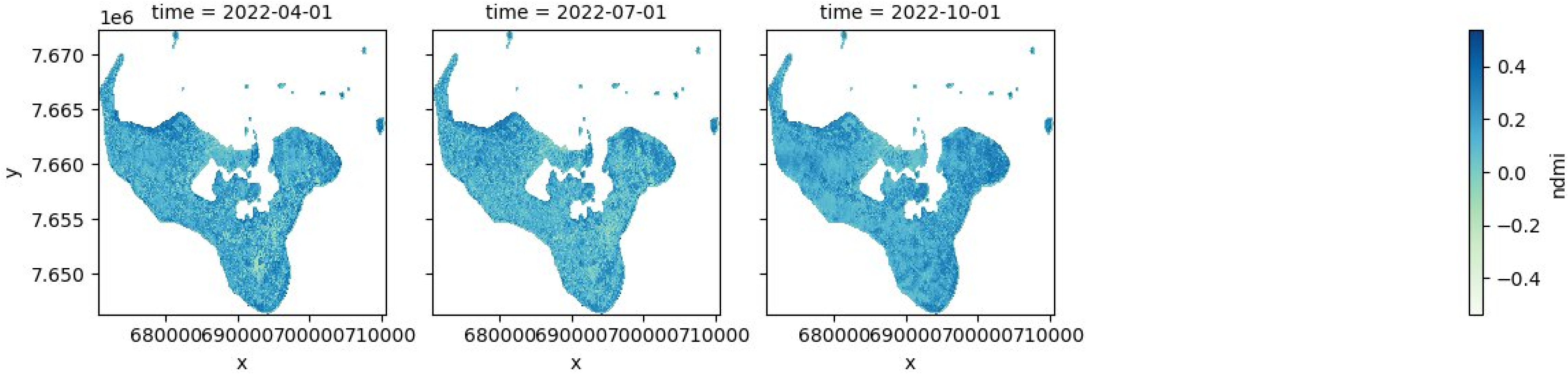
Generate Normalised Difference Moisture Index

Computes Normalized Difference Moisture Index.

Used to determine vegetation water content. For drought monitoring.

```
data = dep.get_ndmi(aoi, timeframe="2022-06-01/2023-07-30", period="monthly")

dep.visualise(data, cmap=”PuBu”)
```

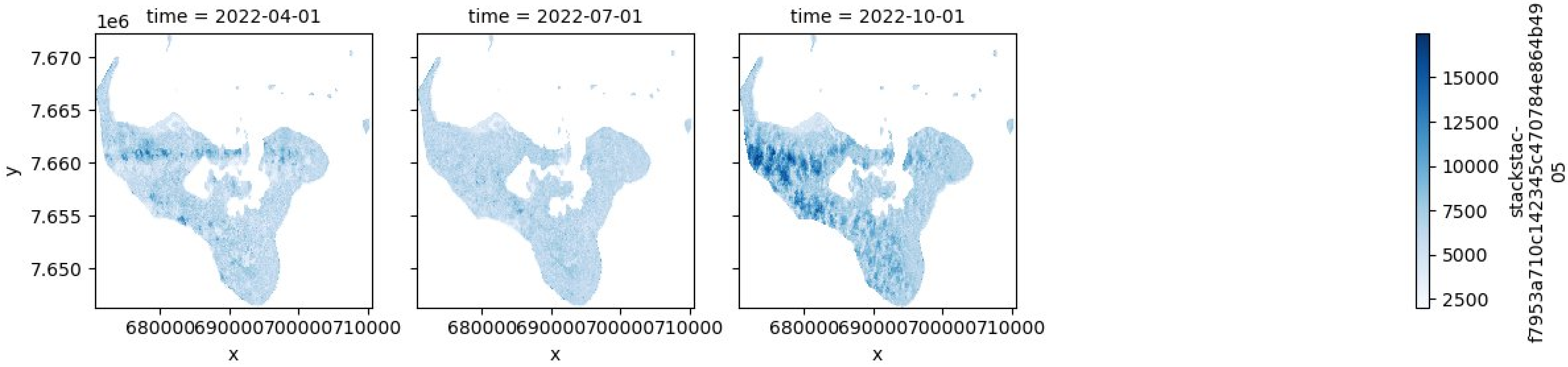


Generate Normalised Difference Water Index

Shows changes in land water bodies, wetlands etc.

```
data = dep.get_ndwi(aoi, timeframe="2022-06-01/2023-07-30", period="monthly")
```

```
dep.visualise(data, cmap=”Blues”)
```

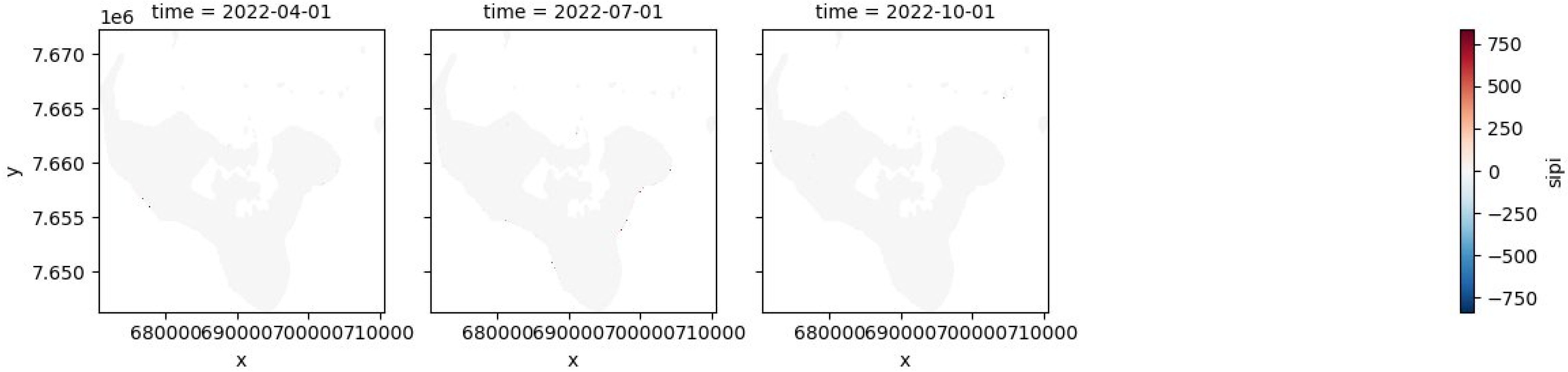


Generate Structural Pigment Index

Computes **Structure Insensitive Pigment Index (SIPI)** which helpful in early disease detection in vegetation, and on set of crop diseases.

```
data = dep.get_sipi(aoi, timeframe="2022-06-01/2023-07-30", period="monthly")
```

```
dep.visualise(data)
```



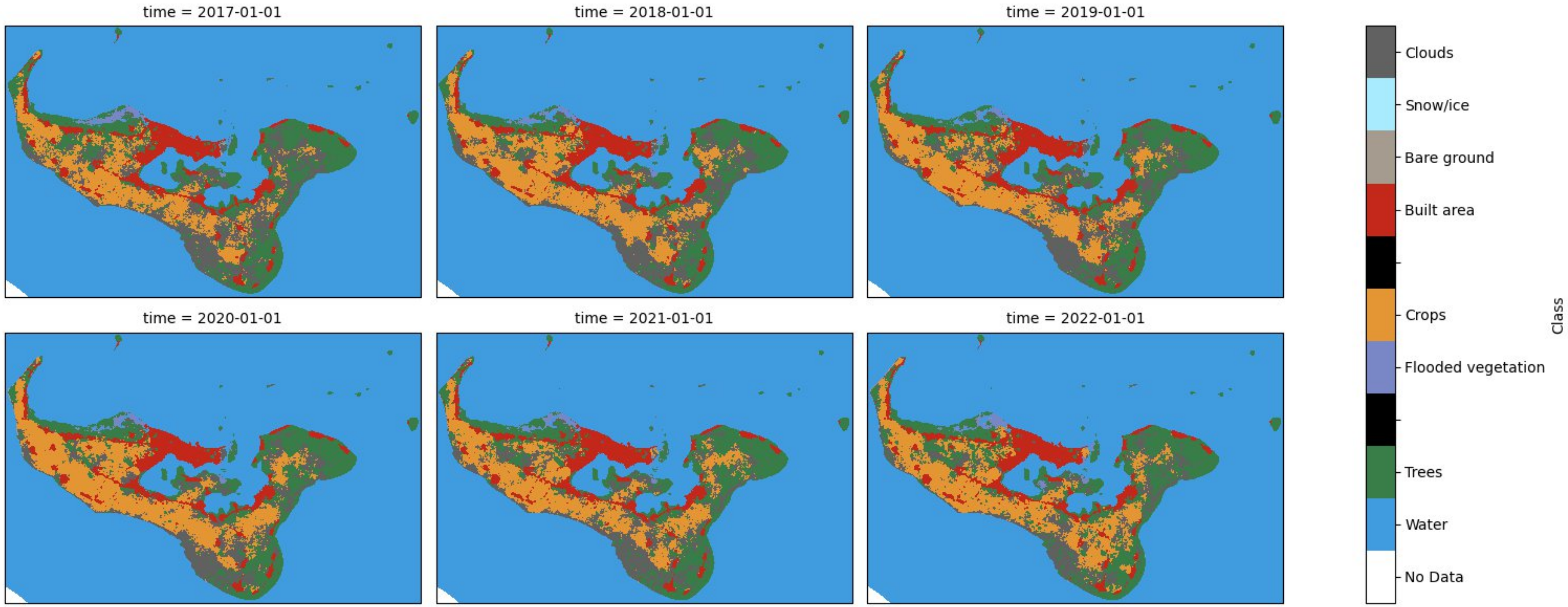
Global LandCover Visualisation

List available Global Land Cover Datasets:

```
data = dep.list_global_land_cover()
```

By default, we will be using ESA 10-class 10m/2 resolution 2017-2022 model.

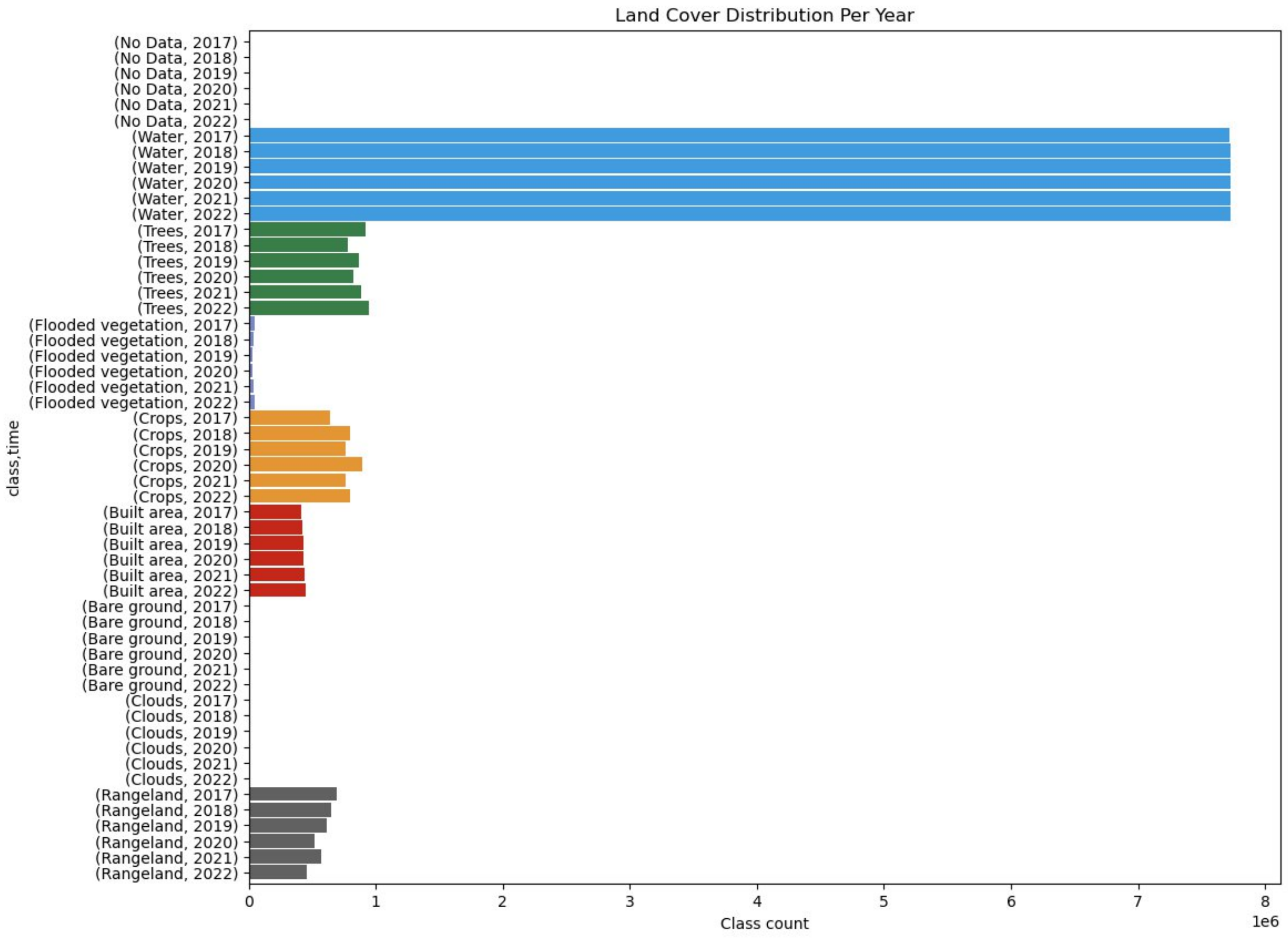
```
dep.chart_global_land_cover(data)
```



Global LandCover Analysis

Undertake change detection across the classes over the model years.

`dep.chart_global_land_cover(data)`



DEPAL GitHub Site

https://github.com/digitalearthpacific/depal